Numerical Computing - CSCI 3656--001
Homework 3 9/17/21
Kai Handelman

1.

$$f(x) = \frac{1}{\exp(x)+1} - \frac{1}{2} = \frac{1}{e^x+1} - \frac{1}{2} = (e^x+1)^{-1} - \frac{1}{2}$$

$$f'(x) = -1 \cdot (e^x+1)^{-2} \cdot e^x$$
$$= -\frac{e^x}{(e^x+1)^2}$$

$$f'(x) = -\frac{e^x}{(e^x+1)^2}$$

2. Please See the function: “newtonMethod()” in the attached script
3. Please See the function: “hwHelper()” in the attached script
4.

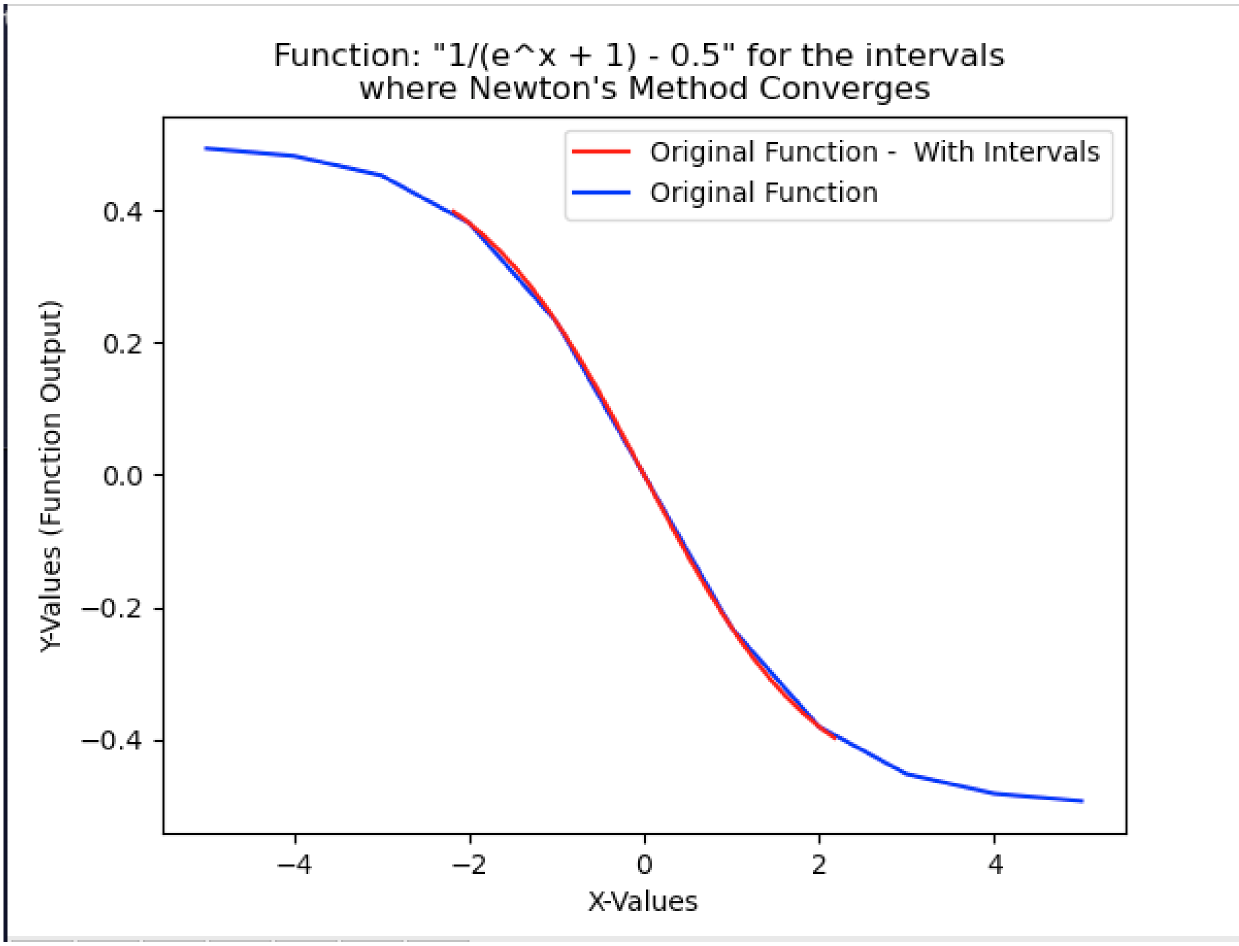
```
Grid:
 Start: -5.0
 End: 5.0
 # of Points: 5000

(Approx.) Lowest intial Guess that's valid is -2.1754350870174033:
 With a Final Guess of: -1.4046813232439052e-16

(Approx.) Highest intial Guess that's valid is 2.1754350870174033:
 With a Final Guess of: 3.6385147282255725e-17

(base) kai@Kais-MacBook-Pro-2 HW3 %
```

Extra Credit:
1.



My educated guess would be, beyond each endpoint (absolute value of approx. 2.17) the next iteration to find x_1 would result in the f'(x_1) being close or is 0. Making it impossible to converge at a root at the point of x_1 or further x's after a couple of iterations. Basically, as more iterations go on the error will grow if the initial guess is larger than the absolute value of (appox.) 2.17

2.

$$f(x) = \frac{1}{(e^x+1)} - \frac{1}{2} \qquad f'(x) = -\frac{e^x}{(e^x+1)^2}$$

root $= 0$ $\qquad f(0)=0$

Since root $=0$, e_n will always equal absolute value of (x_n)

Checking the ratio between e_{n+1} & e_n for intial Guess

$$\frac{e_1}{e_0} = \left|\frac{x_1}{x_0}\right| = \frac{x_0 - \frac{f(x_0)}{f'(x_0)}}{x_0}$$

if this ratio is ≥ 1 we know any error we start with will strictly grow for the next iteration

Basically, I want to find an initial guess x_0 where the magnitude of e_0 is strictly less than e_1. By finding this "tipping point" I can conclude that any initial guess that breaks this will result in the newton's method not to converge. Because of how I this up, I can also conclude that any number greater (in magnitude) from that tipping point initial guess will result in an error iteration of strictly larger or equal, meaning that the method won't converge to the root.

Simplifying the ratio:

$$\frac{e_1}{e_0} = \left| \frac{x_0 - \frac{f(x_0)}{f'(x_0)}}{x_0} \right|$$

$$= \left| \frac{x_0 + \frac{\frac{1}{e^{x_0}+1} - \frac{1}{2}}{e^{x_0} \cdot \frac{1}{(e^{x_0}+1)^2}}}{x_0} \right|$$

$$\frac{f(x_0)}{f'(x_0)} = -\frac{\frac{1}{e^{x_0}+1} - \frac{1}{2}}{e^{x_0} \cdot \frac{1}{(e^{x_0}+1)^2}}$$

$$= -\frac{\left(\frac{1}{e^{x_0}+1} - \frac{1}{2}\right)}{e^{x_0}} \cdot (e^{x_0}+1)^2$$

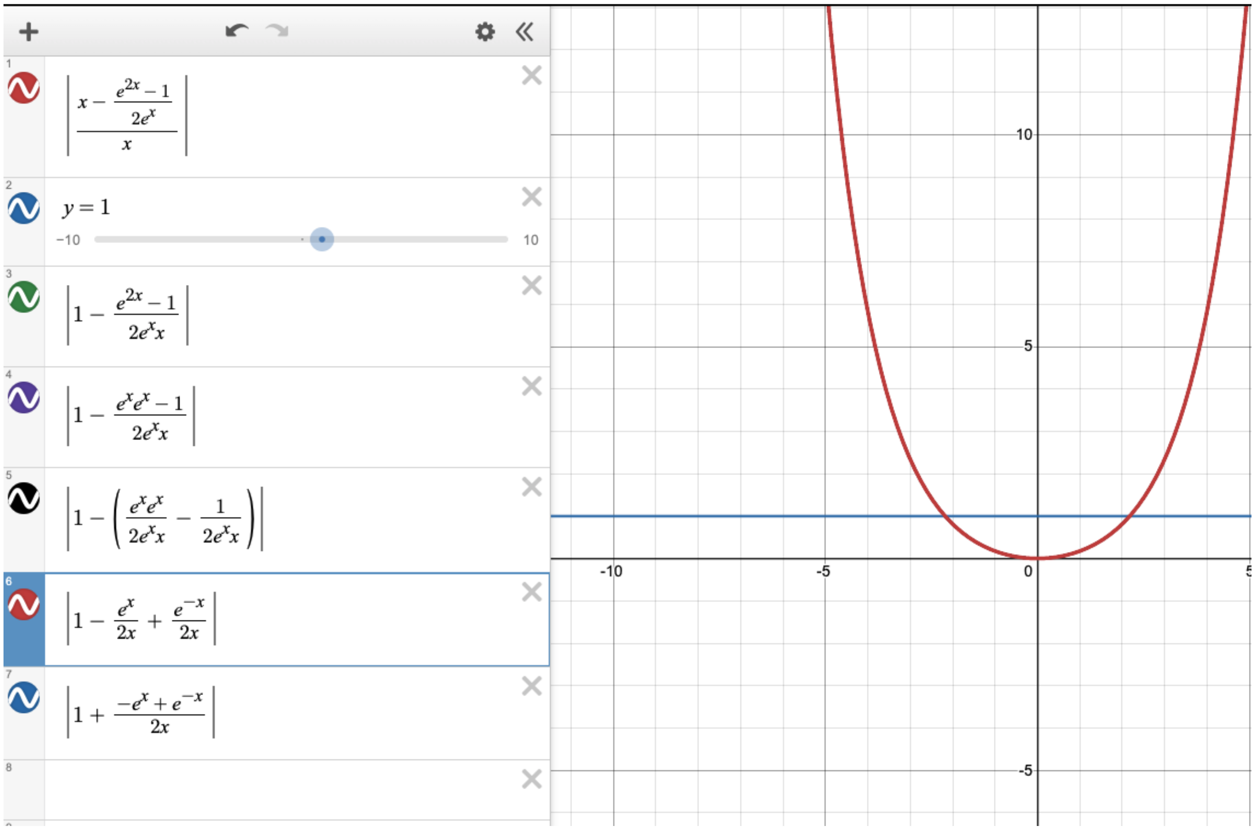
$$= \frac{(e^{2x_0}-1)}{2e^{x_0}}$$

$$= \left| \frac{x_0 - \frac{e^{2x_0}-1}{2e^{x_0}}}{x_0} \right|$$

I’ve spent a lot of time trying to simplify the equation further than this to easily identify what the tipping point would be, however, I kept running into some sort of algebraic errors.

I’ve resulted in plotting this on Desmos and comparing it to y = 1 to find where the intersections are.

Please see below:



The intersection points resulted in x = 2.177,-2.177, which are very close to the 2.175... and -2.175… points I found in problem 4.

Code: Zip at
https://drive.google.com/drive/folders/1PAD9vINXj7bakwce2TWkXd3ydnAE2rFC?usp=sharing

hwScript.py

Users > kai > Desktop > Fall 2021 > Numerical Computations > HW3 > hwScript.py

```python
import math
import numpy as np
import matplotlib.pyplot as plt

def newtonMethod(func, derFunc, tol, guess,root):
    cur = guess
    if derFunc(root) == 0:
        return None
    else:
        cur = guess
        while abs(cur - root) > tol:
            try:                                        #Check if the derivate function at x_n is 0
                cur = cur - (func(cur)/derFunc(cur))
            except:
                return None
    if cur != cur:                                      #Check cur ends up at nan
        return None
    return cur

def hwHelper(showVisuals):
    func = lambda x: 1/(np.exp(x)+1) - 1/2                #Function Set Up
    dFunc = lambda x: -np.exp(x)/math.pow(np.exp(x)+1,2)  #First Derivative
    tol = 10**(-9)                                        #Tolerance
    root = 0                                              #Root of Funciton

    start = -5.0                                          #Variables for the interval [-5,5]
    end = 5.0
    points = 50000
    potentialGuess = np.linspace(start,end,num=points)    #Created 5000 points between -5 and 5


    valids = []
    for x in potentialGuess:                              #Iterating through each point in potentialGuess and saving all the points that coverged into an array
        if newtonMethod(func,dFunc,tol,x,root) is not None:
            valids.append((x, newtonMethod(func,dFunc,tol,x,root)))


    # Visuals - Printing out all the results
    if showVisuals:
        print("\n\n\n\n\n")

        print("Grid: \n Start: {}\n End: {} \n # of Points: {}\n".format(start,end,points))

        print("(Approx.) Lowest intial Guess that's valid is {}:\n With a Final Guess of: {}\n".format(valids[0][0],valids[0][1]))
        print("(Approx.) Highest intial Guess that's valid is {}:\n With a Final Guess of: {}\n".format(valids[len(valids)-1][0],valids[len(valids)-1][1]))
    return (valids[0][0],valids[len(valids)-1][0])


def extraCreditOne():
    orignalFunc = lambda x: 1/(np.exp(x)+1) - 1/2
    intevals = hwHelper(True)
    x = np.linspace(intevals[0],intevals[1],num=1000)
    y = orignalFunc(x)


    xw = np.arange(-5,6)
    yw = orignalFunc(xw)
    line2, = plt.plot(xw,yw,color = "blue",label="Original Function")
    line1, = plt.plot(x,y,color = "red",label="Original Function -  With Intervals")
    plt.xlabel("X-Values")
    plt.ylabel("Y-Values (Function Output)")
    plt.title("Function: \"1/(e^x + 1) - 0.5\" for the intervals \nwhere Newton's Method Converges")
    plt.legend(handles=[line1,line2])
    plt.show()


# hwHelper(True)
extraCreditOne()
```

Python 3.7.3 64-bit ⊗ 0 ⚠ 0 kai-handelman Live Share